JC10 Rec'd PCT/PTO 21 DEC 2001

FORM PTO-1390 (Modified) (REV 5-93) U.S. DEPARTMENT OF COMMERCE PATENT AND TRADEMARK OFFICE TRANSMITTAL LETTER TO THE UNITED STATES DESIGNATED/ELECTED OFFICE (DO/EO/US) CONCERNING A FILING UNDER 35 U.S.C. 371		ATTORNEY'S DOCKET NUMBER 017346-0172
		U.S. APPLICATION NO. (If known, see 37 C.F.R. 1.5) Unassigned 10/018780
INTERNATIONAL APPLICATION NO. PCT/FR00/01767	INTERNATIONAL FILING DATE 06/23/2000	PRIORITY DATE CLAIMED 06/25/1999

TITLE OF INVENTION
DEVICE FOR MANAGING DATA EXCHANGES BETWEEN DATA PROCESSING EQUIPMENT

APPLICANT(S) FOR DO/EO/US
Fabrice ROUILLIER and Jean-Charles FAUGERE

Applicant herewith submits to the United States Designated/Elected Office (DO/EO/US) the following items and other information:

1. ☒ This is a FIRST submission of items concerning a filing under 35 U.S.C. 371.
2. ☐ This is a SECOND or SUBSEQUENT submission of items concerning a filing under 35 U.S.C. 371.
3. ☐ This express request to begin national examination procedures (35 U.S.C. 371(f)) at any time rather than delay examination until the expiration of the applicable time limit set in 35 U.S.C. 371(b) and PCT Articles 22 and 39(1).
4. ☒ A proper Demand for International Preliminary Examination was made by the 19th month from the earliest claimed priority date.
5. ☒ A copy of the International Application as filed (35 U.S.C. 371(c)(2))
   - ☐ is transmitted herewith (required only if not transmitted by the International Bureau).
   - ☒ has been transmitted by the International Bureau.
   - ☐ is not required, as the application was filed in the United States Receiving Office (RO/US)
6. ☐ A translation of the International Application into English (35 U.S.C. 371(c)(2)).
7. ☒ Amendments to the claims of the International Application under PCT Article 19 (35 U.S.C. 371(c)(3))
   - ☐ are transmitted herewith (required only if not transmitted by the International Bureau).
   - ☐ have been transmitted by the International Bureau.
   - ☐ have not been made; however, the time limit for making such amendments has NOT expired.
   - ☒ have not been made and will not be made.
8. ☐ A translation of the amendments to the claims under PCT Article 19 (35 U.S.C. 371(c)(3)).
9. ☐ An oath or declaration of the inventor(s) (35 U.S.C. 371(c)(4)).
10. ☐ A translation of the annexes to the International Preliminary Examination Report under PCT Article 36 (35 U.S.C. 371(c)(5)).
11. ☐ Applicant claims small entity status under 37 CFR 1.27 .

Items 12. to 17. below concern other document(s) or information included:

12. ☐ An Information Disclosure Statement under 37 CFR 1.97 and 1.98.
13. ☐ An assignment document for recording. A separate cover sheet in compliance with 37 CFR 3.28 and 3.31 is included.
14. ☐ A FIRST preliminary amendment.
    ☐ A SECOND or SUBSEQUENT preliminary amendment.
15. ☐ A substitute specification.
16. ☐ A change of power of attorney and/or address letter.
17. ☐ Other items or information:

JC03 Rec'd PCT/PTO 21 DEC 2001

U.S. APPLICATION NO. (If known, see 37 C.F.R. 1.50)	INTERNATIONAL APPLICATION NO	ATTORNEY'S DOCKET NUMBER
Unassigned 10/018780	PCT/FR00/01767	017346-0172

18. ☒ The following fees are submitted:	CALCULATIONS	PTO USE ONLY
Basic National Fee (37 CFR 1.492(a)(1)-(5): Search Report has been prepared by the EPO or JPO.....$890.00		
International preliminary examination fee paid to USPTO (37 CFR 1.482).....$710.00		
No international preliminary examination fee paid to USPTO (37 CFR 1.482) but international search fee paid to USPTO (37 CFR 1.445(a)(2) $740.00		
Neither international preliminary examination fee (37 CFR 1.482) nor International search fee (37 CFR 1.445(a)(2)) paid to USPTO $1,040.00		
International preliminary examination fee paid to USPTO (37 CFR 1.482) and all claims satisfied provisions of PCT Article 33(2)-(4) $100.00		
ENTER APPROPRIATE BASIC FEE AMOUNT =	$890.00	
Surcharge of $130.00 for furnishing the oath or declaration later than 20 Months from the earliest claimed priority date (37 CFR 1.492(e))	$130.00	

Claims	Number Filed		Included in Basic Fee		Extra Claims		Rate		
Total Claims	14	-	20	=	0	×	$18.00	$0.00	
Independent Claims	2	-	3	=	0	×	$84.00	$0.00	
Multiple dependent claim(s) (if applicable)							$280.00	$280.00	

TOTAL OF ABOVE CALCULATIONS =	$1300.00	
Reduction by ½ for filing by small entity, if applicable.	$0.00	
SUBTOTAL =	$1300.00	
Processing fee of $130.00 for furnishing English translation later the 20 months from the earliest claimed priority date (37 CFR 1.492(f). +	$130.00	
TOTAL NATIONAL FEE =	$1430.00	
Fee for recording the enclosed assignment (37 CFR 1.21(h)). The assignment must be accompanied by an appropriate cover sheet (37 CFR 3.28, 3.31). $40.00 per property +		
TOTAL FEES ENCLOSED =	$1430.00	

				Amount to be: refunded	$
				charged	$

a. ☒ A check in the amount of $1430.00 to cover the above fees is enclosed.

b. ☐ Please charge my Deposit Account No. 19-0741 in the amount of $1430.00 to the above fees. A duplicate copy of this sheet is enclosed.

c. ☒ The Commissioner is hereby authorized to charge any additional fees which may be required, or credit any overpayment to Deposit Account No. 19-0741. A duplicate copy of this sheet is enclosed.

NOTE: Where an appropriate time limit under 37 CFR 1.494 or 1.495 has not been met, a petition to revive (37 CFR 1.137(a) or (b)) must be filed and granted to restore the application to pending status.

SEND ALL CORRESPONDENCE TO:

Foley & Lardner
Customer Number: 22428

22428
PATENT TRADEMARK OFFICE

Brian McNamara
SIGNATURE

NAME BRIAN J. MCNAMARA

REGISTRATION NUMBER 32,789

Rec'd PCT/PTO 11 MAR 2002
10/018780

DEVICE FOR MANAGING DATA EXCHANGES BETWEEN DATA PROCESSING EQUIPMENT

The invention relates to the field of data exchange between data processing equipment of identical or nonidentical types, such as software, microprocessors, databases, and the like.

In the data processing field, the use of ever more powerful microprocessors allows an ever greater reduction in computation time or processing time. However, this entails a manipulation, in real time, of an ever bigger number of data items, for example greater than 1 Gb (giga-bytes) in the case of systems such as FGB and RS, or in the case of database management systems (DBMS) and image processing systems.

In a microprocessor, or more generally in data processing equipment, the data items are generally stored in the form of "packets" of k bits (binary information) in registers with a capacity of (n * k) bits. Here, the term register is understood to mean floating registers, certain memory components of microprocessors, graphics cards, MMXs and the like. A microprocessor is therefore unable to process data items (integers) whose size is greater than the capacity of its registers, i.e. (n * k) bits.

In general, each "packet" comprises k = 8 bits. One then speaks of a byte. For example, the

largest integer which a 32-bit microprocessor can process comprises 4 bytes ($n = 4$, $k = 8$).

The order in which the packets of k bits (for example bytes) are stored often varies from one machine to another. The items of equipment are then said to exhibit different internal codings, or stated otherwise arrangements of packets of k bits which are different.

Now, regardless of what the machine is, the binary representation of a datum object (for example an integer) of size less than k bits is invariant (all the bits are given in the same order). Such a representation is therefore common to everyone.

A real problem therefore arises when two items of equipment operating according to different internal codings wish to exchange data objects (scalar) whose dimensions are greater than or equal to k bits. This problem is further heightened when the dimensions ($k * n$) of the registers of the items of equipment differ.

By way of example, the integer 33,751,553 which decomposes in the base $\{2^8\}$ into the form $1 + 2*2^8 + 3*2^{16} + 2*2^{24}$ is coded in an ALPHA or PC type microprocessor by the string of $n = 4$ coefficients [1,2,3,2]. Now, in a SPARC type microprocessor this integer is coded by the string of $n = 4$ coefficients [2,3,2,1] which for a microprocessor designates the integer 16,909,058 ($2 + 3*2^8 + 2*2^{16} + 2^{24}$).

In this example, it is appreciated that a

permutation of the coefficients differentiates between the two internal codings.

In order to allow such items of equipment to exchange their integers, it is therefore vital for them to know their respective internal codings, or stated otherwise the permutations which will allow them to transform their respective codings.

Now, at present, permutations are given as a function of pairs (k,n) which are fixed once and for all, generally with the aid of software such as XDR (registered trademark of the company SUN).

Such software in fact ensures the transcription of machine integers and floating integers which can be coded on 8, 16 and 32 bits (a convention is proposed for the integers of 64, but not beyond) into an external coding (or transmission coding) which turns out to be identical to the internal coding of microprocessors of SPARC type. The external coding can be dubbed "common or universal language". In this software, the arrangement of the k bits of each packet (byte for k = 8) is always invariant.

This type of transcription requires, for each data exchange, a first conversion (or encoding) of the first internal code of the "sending" item of equipment to the external coding, then a second conversion (or decoding) of the external code to the second internal code of the receiving item of equipment.

The double conversion is also performed when the items of equipment are mutually compatible (same internal coding incompatible with an external coding of XDR type). To avoid this it is of course possible to reconfigure the XDR software, but this entails manipulation by an operator.

Moreover, in the current state of matters, the XDR software is difficult to use in 64-bit environments, and is unusable in 128-bit environments. More generally, as soon as scalar data items exceed 32 bits, the XDR software leaves the steering of operations ("the hand") to the user when processing data items greater than 32 bits. Furthermore, it is not designed to function with packets different from 8 bits.

The known solutions therefore do not allow dynamic parameterization and/or exchanges between items of equipment independently of their respective architectures.

To summarize, no known solution affords complete satisfaction as regards speed, efficiency and adaptability.

The aim of the invention is therefore to improve the situation as regards data exchange.

To this end it proposes a device intended to work on primary elementary data items individually coded according to a first arrangement of words (or internal coding), and comprising:

* storage means where first and second sets of symbols are stored, all different, forming, respectively, a representation of the first arrangement and a second arrangement of words (or external coding), a priori different from the first, and

* an operator able to receive as input the first and second sets of symbols and a primary elementary data item, such as an integer, so as to perform on the latter word transformations defined solely by the first and second sets of symbols in such a way as to output a corresponding secondary data item equivalent to the primary elementary data item.

A completely parameterizable and dynamic converter is thus made available.

The invention finds a particularly beneficial application when a first and a second item of equipment wish to exchange primary elementary data items. In this case, the first item of equipment delivers primary elementary data items coded according to the first arrangement (or first internal coding), while primary elementary data items coded according to a fourth arrangement (second internal coding) by a second item of equipment are converted by means of conversion into the form of secondary data items coded according to a third arrangement (second external coding).

As indicated in the introduction, the word arrangement should be regarded here as a layout of

groups of bits in a register (in practice, each group generally being formed of 8 bits (or byte)).

According to the invention, the operator of the device comprises means of interrogation which perform the following operations:

* firstly, they supply the second item of equipment with a message which contains the second set of symbols and requires the sending back of a primary elementary data item, transform of the second set of symbols by the coding according to the fourth arrangement;
* then, they deduce from this primary elementary data item and from the first and second sets of symbols a third set of symbols forming a representation of the fourth arrangement;
* thereafter, they replace the second set of symbols by the third set of symbols, both in the operator and in the means of conversion, so that:
    - in the event of the transmission of a primary elementary data item coded according to the first arrangement and intended for the second item of equipment, the operator delivers to it, directly, a primary elementary data item coded according to the fourth arrangement, and
    - in the event of the transmission of a primary elementary data item coded according to the fourth arrangement and intended for the first item of equipment, the operator delivers to it, directly, a primary elementary data item coded according to

the first arrangement.

In this way, in particular when the items of equipment are of radically different types, and consequently exhibit different internal codings (first and fourth arrangements) and external codings (second and third arrangements), the device according to the invention can be configured independently of the architectures of the items of equipment wishing to exchange data items.

Once the device has set up a direct link (that is to say has effected a "direct conversion operator" between the first and second internal codings), the time required for the exchange of data items between the items of equipment is very appreciably reduced.

The invention also relates to the processes which will be described hereinbelow and which allow the device to ensure their conversions.

Other characteristics and advantages of the invention will become apparent on examining the detailed description hereinbelow, and the appended drawings, in which:

- Figure 1 is a diagram illustrating an embodiment of the invention in an application for the exchanging of data items between two computers (or more generally to data processing items of equipment) of different types; and

- Figure 2 is an algorithm describing an embodiment of the process according to the invention.

The appended drawings are, in essence, of definite character. Consequently, they will be able not only to serve to supplement the latter, but also to contribute to the definition of the invention as the case may be.

Reference is firstly made to Figure 1 in order to take stock of the state of matters before the present invention as regards data exchange between two data processing items of equipment, such as computers (or work station) M1 and M2.

Of course, it could relate, more simply, to microprocessors, or even to different software or to databases, possibly installed in one and the same machine (or computer), but functioning according to different internal and/or external codings.

It is important to note that Figure 1 does not represent the prior art as such, but that it makes it possible to depict the elements

In the example illustrated in Figure 1, the computer M1 comprises a microprocessor 1, for example of 32-bit SPARC type. This microprocessor 1 is implanted on an electronic card so as to be able to cooperate with a hard disk 2.

The microprocessor 1, under the control of the operating system 3 of the computer, stored on the hard disk 2, performs operations on data items, and

delivers on an output 4, 32-bit data items (when it is of SPARC type) according to a first arrangement (or first internal coding).

Each 32-bit data item is then delivered at the output 4 of the microprocessor 1 in the form of an ordered string of 4 bytes (8 bits). The ordered string of four (n) bytes (k = 8) representing an integer E in a 32-bit SPARC microprocessor is called the internal code $M1_{k,n}(E)$. With such a SPARC microprocessor, the internal coding of a 32-bit integer is the string of coefficients of its decomposition in base $\{2^k\}$, ordered according to descending powers.

By way of example, in a PC or ALPHA microprocessor, the internal coding of a 32-bit integer is the string of coefficients of its decomposition in base $\{2^8\}$, ordered according to ascending powers.

To allow the transmission of such integers E, from the computer M1 to the computer M2, it is necessary for the integers to exhibit one and the same format, or external coding. Such is not always the case, as will be seen later.

Consequently, there is provided, conventionally, a conversion module 5, generally installed in the form of software (or a program) on the hard disk of each computer. Of course, the conversion module may be embodied in the form of an electronic circuit.

Be that as it may, this conversion module 5 is linked to an interface 6 coupled, for example by a wire link, to the interface 6′ of the computer M2 with which it wishes to exchange data items.

To define the external coding (or second arrangement), we call upon a base $\{1, 2^k, 2^{2k}, \ldots, 2^{nk}\}$, where k and n respectively designate the number of bits of each word of the arrangement and the number of elements of the base, and where n*k is equal to the number of bits of the data item.

For example, in the aforesaid case, the first arrangement is defined by n = 4 words of k = 8 bits, i.e. n*k = 32 bits.

In this example, the function of the conversion module 5 is therefore to convert a primary elementary data item delivered by the output 4 of the microprocessor 1 (that is to say supplied according to the first internal coding or first arrangement $M_{1,k,n}$) into a secondary data item coded according to the external coding, or second arrangement, $D_{1,k,n}$.

Generally, what differentiates the internal coding from the external coding, in one and the same item of equipment, is a permutation type operation. In this case, we have the relation:

$$D_{1,k,n}(E) = \Phi_{1,k,n}(M_{1,k,n}(E))$$

It is quite obvious that, in certain cases, the permutation $\Phi_{m,k,n}(m=1,2)$ may be the identity. In this case, M and D consist of the same ordered string

of words defining the integer. They are then said to exhibit the same arrangement, or format.

What has just been stated in respect of the first computer M1 applies equally to the second computer M2. Only the internal coding $M_{2,k,n}$, as well as possibly the external coding $D_{2,k,n}$, are different. Here, the expression different should be understood to mean either arrangements (or strings) whose elements (or words) are ordered differently, or arrangements which do not exhibit the same number of elements (k1 and k2 different and/or n1 and n2 different).

Generally, especially in a client/server type environment, the items of equipment are substantially homogeneous, so that the external codings which they use are identical. In this case, we have the following relation:

$$\Phi_{2,k,}n(m_{2,k,n}(E)) = D_{k,n}(E) = \Phi_{1,k,n}(M_{1,k,n}(E))$$

In conventional machines, for each primary elementary data item delivered by the microprocessor 1, on its output 4, and coded according to the first arrangement (or first internal coding), the conversion module 5 performs an encoding (or first conversion) intended to supply the interface 6 with a secondary data item (generally an integer E) coded according to the second arrangement (or first external coding). Stated otherwise, available at the output of the conversion means 5 is the following data item:

$$D_{k,n}(E) = \Phi_{1,k,n}(M_{1,k,n}(E))$$

This secondary data item is then addressed to the second computer M2, which will decode it (second conversion) with the aid of its conversion module 5' ($M_{2,k,n}(E) = \Phi^{-1}_{2,k,n}(D_{k,n,}(E))$), so as to supply the microprocessor 1' with a primary elementary data item coded according to its second internal coding (or fourth arrangement), so that it can process this data item.

The same holds when the second computer M2 wishes to transmit a primary elementary data item to the first computer M1, and in particular to its microprocessor 1. The encoding then consists in forming the secondary data item:

$D_{2,k,n}(E) = \Phi_{2,k,n}(M_{2,k,n}(E))$

and the decoding of the secondary data item supplied by the second computer M2 is performed in the conversion module 5 of the first computer M1. This which supplies a primary elementary data item according to the first arrangement (or first internal coding):

$M_{1,k,n}(E) = \Phi^{-1}_{1,k,n}(D_{k,n,}(E))$

The conversion module 5 of each item of equipment Mi (i = 1,2) can perform just a single and unique coding of the internal coding to the external coding, and vice versa. It follows that this type of item of equipment can function only with items of equipment exhibiting at least one common external coding $D_{k,n}$.

Those skilled in the art have proposed that

data exchange software be installed in certain items of equipment, for example on their hard disk. Mention will be made, for example, of the XDR software (registered trademark) from the manufacturer of SUN workstations. This software proposes a library of encoding/decoding functions which allow a first item of equipment of a first type, to exchange data items with a second item of equipment of a second type, once the respective types of these items of equipment have been declared. Consequently, this involves a purely static type conversion operation, since it requires the intervention of an operator knowing the respective types of the two items of equipment.

Furthermore, for each data item to be exchanged, this type of software systematically performs a double conversion. The first conversion consists of the encoding of the primary elementary data item according to the first internal coding into a secondary data item according to the external coding. The second conversion consists of a decoding of the secondary data item according to the external coding into a primary elementary data item according to the second internal coding.

Moreover, unless reconfigured by hand, by a specialist, the XDR software continues to perform its double conversion when the two items of equipment are identical.

This double conversion vastly slows down (at least by a factor of two (2)) the data processing speeds.

The invention comes to afford a solution to this drawback.

In what follows, the invention will be described, with reference to Figures 1 and 2, in an application for the exchanging of data items between two data processing items of equipment, such as computers (or workstation) M1 and M2.

Of course, it could relate, more simply to microprocessors, or even to different software, possibly installed in one and the same machine (or computer), but functioning according to different internal codings.

The invention proposes a device comprising a first part which replaces the conversion module 5 in the first item of equipment M1, and a second part which supplements the first and is installed at least partially in the first item of equipment M1 (the remainder then being installed in the second item of equipment M2).

Preferably, the device is embodied in the form of software modules installed on the hard disk(s). However, it may also be embodied in the form of electronic circuits. A combination of the two (software and circuit) may also be envisaged.

In the first computer M1, are installed storage means capable of storing a first set (or string) of symbols, all different, forming a representation of the first arrangement (or first internal coding). Such storage means are, for example, embodied in the form of lines of programs which return to addresses of registers or memories of the hard disk 2 of the computer M1.

The storage means 7 also store a second set (or string) of symbols, all different, forming a representation of the second arrangement of words (or first external coding), generally different from the first arrangement. In certain cases it may in fact be identical.

Preferably, the sets of symbols consist of an ordered string of n components which characterize, as indicated hereinabove, the first and second arrangements.

The device furthermore comprises, an operator 8 coupled to the storage means 7, as well as to the output 4 of the microprocessor 1. It can thus receive on an input, the first and second sets of symbols, as well as each primary elementary data item coded according to the first arrangement by the microprocessor 1.

This operator 8 is preferably embodied in the form of a software module (or program) calling upon a library of mathematical calculations. Its function is

to perform on the primary elementary data item, received in the form of an ordered string of words, word transformations defined, solely, on the basis of the first and second sets of symbols. Delivered at the output of this operator 8 is a secondary data item equivalent to the primary elementary data item received. The word transformation should be understood, here, within its mathematical definition, that is to say, as a function or mapping.

In this way, a fully configurable tailored conversion module is produced, adaptable to any type of item of equipment.

The device according to the invention furthermore makes it possible to very considerably accelerate the speed of data exchange between two items of equipment M1 and M2 of different types, in particular. In what follows, not only will the internal codings of the two items of equipment M1 and M2 be regarded as different, but their external codings will also be regarded as different. For example, the microprocessor 1 of the computer M1 is of 32-bit SPARC type, while the microprocessor 1' of the second computer M2 is of 64-bit ALPHA type.

To allow these two items of equipment M1 and M2 to communicate, the device according to the invention deploys an interrogation protocol. This protocol is initiated by the first computer M1 with the aim of determining the internal coding (or fourth

arrangement) of the microprocessor 1' of the second computer M2.

The interrogation is carried out by an interrogation module 9, which consists of a software module (or program). This interrogation software module is, preferably, installed in respect of one part 9-1 on the hard disk of the computer M1, and in respect of another complementary part 9-2 on the hard disk 2' of the computer M2. Of course, as indicated previously, each part of the interrogation module 9-1 and 9-2 may be embodied in the form of electronic circuits.

Moreover, it is possible to envisage a variant in which the entire interrogation module 9 is installed on the first computer M1. In this case, the interrogation software module 9 is designed in such a way as to spontaneously install, upon a first interrogation, a few chosen lines of program (which are substantially equivalent to 9-2) on the hard disk 2' of the second computer M2, and preferably in its conversion module 5', when the latter is embodied in the form of a software module.

A mode (or process) of functioning of the device according to the invention will now be described in which the external coding $D_{k,n}$ is assumed to be fixed in advance and identical for both computers. This mode makes it possible to emulate the mode of functioning of the XDR software.

According to the invention, the set of second symbols (representative of the second arrangement) is an ordered string composed of mutually different elements: [1, 2, 3, ..., n] and equal to $D_{k,n}(E)$. This ordered string is associated with the integer $E = 1 + 2*2^k + 3*2^{2k} + \ldots + n*2^{(n-1)*k}$, which will be addressed to the second computer M2 so as to determine its fourth arrangement.

For reasons of convenience, it is assumed here that n is less than 2^k, it being easy to deal with the general case by compounding the processes (or modes of functioning).

Stated otherwise, by addressing the integer E indicated hereinabove, or more exactly by addressing the ordered string [1, 2, 3, ..., n] comprising numbers which are all mutually different, we will, in return, deduce therefrom the second internal code (or fourth arrangement) of the second machine M2.

By way of example, we take k = 8 and $k*n \geq 32$.

To completely determine the permutation $\Phi_{2,k,n}$ deployed by the conversion module 5' of the second computer M2, it is sufficient to take the integer $E = 1 + 2*2^8 + 3*2^{16} + 4*2^{24}$ and the secondary data item $D_{k,n}(E)$ formed of the string of symbols [1, 2, 3, 4].

In practice, the determination of the permutation $\Phi_{2,k,n}$ amounts to constructing an array

denoted, for fixed k, permut, such that permut[n] is the ordered list representing $\Phi_{m,k,n}$.

The program for detection of the internal coding and the construction of the external coding is given by way of example, in the C language, in the module Mod2 of the appendix.

The functions construct ordered lists external_coding[i] and internal_coding[i], for values of i taking the sizes of the usual integers of the C language (char, short, int, and long). It is recalled that in the C language (which is merely one example of a usable programming language) the long variable indicates the maximum size of the registers used by the microprocessor to store the scalar data items. The external_coding and internal_coding variables respectively represent the external coding $D_{k,n}$ and the internal coding $M_{k,n}$. These variables are given by way of example in the module Mod1 in the appendix, where "B8" is the invariant type which makes it possible to represent an integer from the value 0(zero) to the value 255.

For a fixed value i, we construct the integer $E = 1 + 2*2^k + \ldots + n*2^{k*(i-1)}$.

The aforesaid ordered lists form arrays. The external_coding[i] array is then equal, for k = 8, to the external coding $D_{8,i}(E) = [1, 2, 3, \ldots, i]$. The internal_coding[i] array is, for its part, equal to

$M_{m,8,i}(E)$ which varies according to the computer (or machine considered).

Next, a set (or multiplicity) of several permutation functions $\Phi_{m,8,i}$ is defined for values of i varying from 1 to sizeof(long) (or in certain cases from 1 to sizeof(longlong). This latter variable defines the integer scalar exhibiting the largest size admissible in the C language; in general it is the largest integer which can be understood by the machine or item of equipment. These permutation functions ensure allowance for integers codable on 8, 16, 32 and 64 bits. They can be easily extended to larger values, in particular 128 bits.

The permutation functions are defined, by way of example, in the module Mod3 of the appendix. It will be noted that the formulation of this program module makes it possible to detect the values of i for which no permutation is necessary.

Next it is necessary to describe encoding/decoding functions which will be used when the parameters of the problem, namely the internal coding, external coding and permutation function, have been determined.

For example, the function making it possible to transform the external coding of an integer (or array of m elements of 8 bits) of given size (sz) into its internal format can be defined by the module Mod4 of the appendix.

As the person skilled in the art will have observed, the module Mod4 describing the encoding/decoding functions proposes generic functions, insofar as, on the one hand, they depend only on the size of the integers to be processed, and that on the other hand, care is taken not to carry out permutations unless this proves to be necessary.

In the same way, the inverse functions which make it possible to transform the internal coding of an integer (array m of element of 8 bits) of given size (sz) into another internal coding (array p of elements of 8 bits) are defined. These inverse functions are defined, by way of example, by the module Mod5 of the appendix. They therefore allow the direct conversion of a primary elementary data item coded according to the first, respectively fourth, arrangement into a primary elementary data item coded according to the fourth, respectively first, arrangement.

With the aid of modules Mod1 to Mod5, a binary data exchange protocol (or process) is produced, the installation of which is totally independent of the respective architectures of the computers (or machines) considered. Of course, this installation is related to the language used, here the C language. However, a transposition to another computer language may easily be obtained.

To operate the exchanges protocol, the first computer M1 opens the communication channel which links

it to the second computer M2 then activates this second computer M2 and sends it the integer E defined by the second set of symbols, here [1, 2, 3, 4], on the chosen communication channel. The second computer M2 reads the integer, transforms it by coding it according to its fourth arrangement (or second internal coding) and returns this transform on the communication channel.

An exemplary program module making it possible to carry out the operations mentioned above, is given in the module Mod6 (as far as the part 9-1 installed in the first computer M1 is concerned) and in the module Mod7 (as far as the part 9-2 installed in the second computer M2 is concerned).

Of course, and as indicated previously, the part 9-2 intended to activate the second computer M2 can be installed remotely by the first computer M1. To do this, it is sufficient for the interrogation module 9 of M1 to address an adaptation of the program module proposed in Mod7.

In the module Mod6, two arguments are called: a first machine name (first computer) and a second machine name (second computer) which has to be activated by the first machine. Moreover, in Mod7, two arguments are also called: a machine name (or computer) and a port.

In these two modules, the functions send_n (nb, bus, buf, n), respectively read_n (nb, bus, buf, n), write, respectively read, an array buf of n*k bits

(k is in practice fixed at the value 8), consisting of machine integers of nb*k bits on a channel denoted bus.

It is clear that these functions will depend on the chosen communication channel (shared memory, files, databases (DBMS), graphics cards, microprocessor, picture and/or sound storage formats (multimedia), sockets and the like). Such functions call upon (in the form of loops) the functions conv_machine_2_prot_UI and reorder_UI, and manage the input/output buffers if necessary.

In the foregoing, the external coding $D_{k,n}$ was assumed to be fixed in advance. It therefore corresponded to an embodiment of the device according to the invention which was especially well suited to the items of equipment which can communicate with one another by reason of one and the same external coding $D_{k,n}$.

However, the device according to the invention can also be adapted to the exchanging of data items between items of equipment exhibiting different external codings. In this case, the device dynamically calculates a common external coding, this amounting to not fixing any data exchange format a priori.

A preferred solution consists in fixing the internal coding of one of the two machines (or computers) as external exchange coding so that one of the two permutations $\Phi_{1,k,n}$ and $\Phi_{2,k',n'}$ used by these two machines is equal to the identity.

The main steps for the dynamic determination of an exchange format (or external coding) according to the invention, will now be described with reference to Figure 2.

Firstly, in a step 10, the microprocessor 1 of the first computer M1 addresses to the device installed, here on the hard disk 2, a data item coded according to the first arrangement (or first internal coding).

This data item being intended for the second computer M2, whose external coding (or third arrangement) and internal coding (or fourth arrangement) are both unknown, there is performed, in a step 20, an initialization of that part 9-1 of the protocol contained in the first computer M1 and in particular in the operator 8. This consists in fixing a default value for $D_{k,n}$. In fact, this default value for $D_{k,n}$ is the ordered string described previously [1, 2, 3, ..., n]. This is then followed, in a step 30 by an initialization of that part 9-2 of the protocol which is located in the second computer M2.

It is clear, as indicated previously, that when the entire interrogation program is installed in the first computer M1, the lines of program 9-2 ensuring the initialization of the second computer M2 have to be transferred to the latter, for example in its conversion module 5'. This step 30 therefore consists in replacing the external coding $D_{2,k',n'}$ (or

third arrangement) of the second computer M2 by the default value $D_{k,n}$ fixed in step 20.

In fact, M1 addresses to M2 an interrogation message containing the second set of symbols which represents the second arrangement, or else one or more variants thereof, comprising a number (numbers) of symbols which is (are) different (greater or less) so that in the event of different exchange formats, one at least of these sets of symbols can be processed by the conversion module 5' of M2. In this case, it is the storage means 7 of M1 which store the different variants of sets of symbols comprising numbers n_i of words which are different and/or of words of number k_i of bits which are different.

The microprocessor 1' of M2 sends back to the conversion module 5', which henceforth codes in the exchange format $D_{k,n}$ imposed by default, a primary elementary data item coded according to its fourth arrangement (or second external coding) $M_{2,k',n'}$. This data item is converted (encoded) according to the external coding $D_{k,n}$ and addressed to M1 on the communication channel.

In a step 40 the interrogation module 9-1 installed in the first computer M1 will read on the communication channel the data item $M_{2,k',n'}$ (transform of the second symbol set $D_{k,n}$) which represents the internal coding of the second computer M2.

In a step 50 the interrogation module 9-1 of the first computer M1 then replaces the default value of the external coding $D_{k,n}$ by the data item received from the second computer M2, namely $M_{2,k',n'}$.

Next, in a step 60, the default value of $D_{k,n}$ supplied during step 30 is replaced in the second computer M2, and in particular in its conversion module 5'.

It may be noted that, owing to the replacement in M2 of $D_{k,n}$ (external coding imposed by default) by $M_{2,k',n'}$ (internal coding of the microprocessor 1'), a permutation $\Phi_{2,k',n'}$ equal to the identity is manifest in the conversion module 5' of M2. Likewise, an external coding identical to the internal coding of M2 is manifest in the "conversion module" 5 of M1, so much so that any primary elementary data item of a microprocessor 1 or 1' encoded according to the first or the fourth arrangement, can be converted directly into a primary elementary data item of the other microprocessor 1' or 1, coded according to the fourth or the first arrangement.

The two machines M1 and M2 are henceforth ready to exchange primary elementary data items directly in a step 70.

A primary elementary data item delivered by the microprocessor 1' of M2, or by the conversion module 5 of M1, will therefore not undergo any conversion in the conversion module 5'. This conversion

module 5' is, as it were, short-circuited. Moreover, a data item intended for the microprocessor 1 of M1 is converted, directly, from the internal format (fourth arrangement) of M2 to the internal format (first arrangement) of M1.

It is clear that the process just described with reference to the algorithm illustrated in Figure 2 adapts automatically, through a phase of negotiation, to the case, described previously, in which the two computers (or machines) originally exhibit the same external coding. In this case, the initialization steps 20 and 30 are unnecessary.

An exemplary program, making it possible to deploy the algorithm illustrated in Figure 2 is given in the module Mod8 of the appendix. With this program are associated, as indicated previously, two programs, of the type of those given in the modules Mod6 and Mod7 of the appendix. These are programs making it possible respectively to activate the conversion module 5 of the first computer (module Mod9 of the appendix) and of the module making it possible to activate the second computer M2 (module Mod10 of the appendix).

In this invention, the definition of the scalar data items must be taken in its broad sense, that is to say as base units. Under these conditions, the concept of coding must likewise be taken in its broadest sense, that is to say as a process (or mode)

intended to order scalar units. The coding may therefore be forced or initiated by a user, as need be.

The invention is not limited to the modes of device implementation and process implementation described hereinabove, merely by way of example, but it encompasses all the variants which may be envisaged by the person skilled in the art within the framework of the claims hereinbelow.

Thus, a device and the associated process have been described, in which the second set of symbols (representative of the second arrangement) consisted of a string of n elements, all different and of values ascending from 1 to n. However, it is clear that any other string of n different elements could be used.

Appendix

* Mod1 :

```
Typedef longlong Ilonglong
Typedef unsigned int UI32

#define _max_size_type 255
B8 external_coding[_max_size_type][_max_size_type];
B8 internal_coding[_max_size_type][_max_size_type];
UI32 permut[_max_size_type][_max_size_type];
```

* Mod2 :

```
void init_permut_char(String tab_int)
{
  tab_int[0]=1;
}

#define genere_permut(typ) \
void init_permut_ ## typ ## (String tab_int) \
  { \
     typ res=1; \
     typ tmp=2; \
     typ inc=256;\
     typ pow=1; \
     UI32 j=1; \
     for(j=1;j<sizeof(typ);j++) { \

        pow*=inc;\
        res+=(tmp)*pow;\
        tmp++; \
    }\
    memcpy((void *)tab_int,(void *)(&res),sizeof(typ));\
  }

genere_permut(short)
genere_permut(int)
genere_permut(long)
genere_permut(Ilonglong)
```

```
void set_permutation_0(String int_coding,const UI32 sz)
{
  if(sz==sizeof(char)){
    init_permut_char(int_coding);
  }
  if(sz==sizeof(short)){
    init_permut_short(int_coding);
  }
  if(sz==sizeof(int)){
    init_permut_int(int_coding);
  }
  if(sz==sizeof(long)){
    init_permut_long(int_coding);
  }
  if(sz==sizeof(Ilonglong)){
    init_permut_Ilonglong(int_coding);
  }
}

void set_permutations_0()
{
  UI32 i=1;
  while (i<=maxsize_prot) {
    set_permutation_0(&(internal_coding[i][0]),i);
    i*=2;
  }
}


/* maxsize_prot defines the maximum size of the integers
   which have to be considered by the protocol */

void set_exchange_0()
{
  UI32 i=1,j;
  while (i<=maxsize_prot) {
    for(j=0;j<i;j++){
       external_coding[i][j]=j+1;
    }
    i*=2;

  }
}
```

* <u>Mod3</u> :

```
UI32 find_elem(B8 elt,String tab,const UI32 sz)
{
  UI32 i=0;
  while ((i<sz) && ((tab[i])!=(elt))) i++;
  return(i);
}

void define_permutation(String int_coding,String ext_coding,
UI32 ** perm_coding,const UI32 sz)
{
  UI32 i;
  if(strncmp(int_coding,ext_coding,sz)){
    permut_type[sz]=NEED_PERMUT;
    for(i=0;i<sz;i++) {
      (*perm_coding)[i]=find_elem(int_coding[i],ext_coding,sz);
    }
  }
  else permut_type[sz]=NO_PERMUT;
}


void C__initProtocol()
{
  UI32 i=1;
  UI32 *ptr;
  set_exchange_0()
  C__initProtocol_0();
  while (i<=maxsize_prot) {
    ptr=&(permut[i][0]);
    define_permutation(&(internal_coding[i][0]),
                       external_coding[i],&ptr,i);

    i*=2;
  }
}
```

```
* Mod4 :
void reorder_UI(const UI32 sz,String m)
{
  UI32 i;
  static B8 tmp[_max_size_type];
  UI32 *permut_tmp=NULL;
  if(permut_type[sz]==NEED_PERMUT) {
    permut_tmp=&(permut[sz][0]);
    for(i=0;i<sz;i++){
      tmp[i]=m[permut_tmp[i]];
    }
    for(i=0;i<sz;i++){
      m[i]=tmp[i];
    }
  }
}

* Mod5 :
void conv_machine_2_prot_UI(const UI32 sz,String p,Cste_String m)
{
  UI32 i;
  UI32 *permut_tmp=NULL;
  if(permut_type[sz]==NEED_PERMUT) {
    permut_tmp=&(permut[sz][0]);
    for(i=0;i<sz;i++){
      p[permut_tmp[i]]=m[i];
    }
  }
  else {
    for(i=0;i<sz;i++){
      p[i]=m[i];
    }
  }
}
* Mod6 :

*/
int main(int argc, char** argv)
{
  UI32 te=1234,re=0;
  /* opens the communication channel and starts up the server */
  BUS GB=createBus(argv[1],argv[2]);
  C__initProtocol();
  send_n(sizeof(UI32),GB,(char *)(&te),1);
  read_n(sizeof(UI32),GB,(char *)(&re),1);
  fprintf(stderr,''%u'',te);
  fprintf(stderr,''%u'',re);
}

/*
```

* Mod7 :

```
*/
int main(int argc, char** argv)
{
  UI32 te=0;
  /* opens the communication channel and starts up the server */
  BUS GB=createBus(argv[1],argv[2]);
  C_initProtocol();
  read_n(sizeof(UI32),GB,(char *)(&te),1);
  te++;
  send_n(sizeof(UI32),GB,(char *)(&te),1);
```

* Mod8 :

```
void C_reinitProtocol()
{
  UI32 i=1;
  UI32 *ptr;
  set_exchange_0()
  C_initProtocol_0();
  while (i<=maxsize_prot) {
    ptr=&(permut[i][0]);
```

* Mod9 :

```
*/
int main(int argc, char** argv)
{
  UI32 te=1234,re=0;
  /* maximum size of integers which is supported by this processor */
  B8 maxsize_loc=sizeof(long);
  /* maximum size of integers which is supported by the client */
  B8 maxsize_ext=0;
  /* opens the communication channel and starts up the server */
  BUS GB=createBus(argv[1],argv[2]);

  /*
     default initialization of the protocol
  */
  C_initProtocol();

  read_n(1,GB,(char *)(&(maxsize_ext)),1);
  send_n(1,GB,(char *)(&(maxsize_loc)),1);
  /*
     calculation of the maximum size of integers which is
     supported by the protocol
  */
```

```
    maxsize_prot=min(maxsize_loc,maxsize_ext);

    /*
      reception of the internal coding for the machine on which
      the server for the integers which is admissible for the
      protocol functions, assignment to the exchange coding
    */
    for(i=1;i<=((UI32)maxsize_prot);i*=2){
      read_n(1,GB,(char *)(&(external_coding[i][0])),i);
    }

    /*
       re-initialization of the protocol with the new
       exchange format
    */
    C__reinitProtocol();

    /* sending of the commands */
    send_n(sizeof(UI32),GB,(char *)(&te),1);

    /* reading of the result */

    read_n(sizeof(UI32),GB,(char *)(&re),1);
    fprintf(stderr,''%u'',te);
    fprintf(stderr,''%u'',re);
}
```

* Mod 10 :

```
*/
int main(int argc, char** argv)
{
   UI32 te=0,i;
   /* maximum size of integers which is supported by this processor */
   B8 maxsize_loc=sizeof(long);
   /* maximum size of integers which is supported by the client */
   B8 maxsize_ext=0;
   /* opens the communication channel and starts up the server */
   BUS GB=createBus(argv[1],argv[2]);

   /*
      default initialization of the protocol
   */
   C__initProtocol();

   send_n(1,GB,(char *)(&(maxsize_loc)),1);
   read_n(1,GB,(char *)(&(maxsize_ext)),1);
```

```
/*
   calculation of the maximum size of integers
   which is supported by the protocol
*/
maxsize_prot=min(maxsize_loc,maxsize_ext);

 /*
   sending of the internal coding for this machine
   of the integers which is admissible for the protocol
 */
 for(i=1;i<=((UI32)maxsize_prot);i*=2){
   send_n(1,GB,(char *)(&(internal_coding[i][0])),i);
 }

 /*
    the internal format of the integers admissible for
    the protocol is assigned to the exchange format
 */

  for(i=1;i<=((UI32)maxsize_prot);i*=2) {
    for(j=0;j<i;j++) {
      external_coding[i][j]=internal_coding[i][j];
    }
  }

  /*
     reinitialization of the protocol with the new
     exchange format
  */
  C__reinitProtocol();

  /* reading of the commands */
  read_n(1,GB,(char *)(&(te)),1);
  te++;
  /* sending back of the result */
  send_n(sizeof(UI32),GB,(char *)(&te),1);
}
```

CLAIMS

1. Data conversion device, intended to work on primary elementary data items individually coded according to a first arrangement of words, characterized in that it comprises:

* storage means (7) for storing a first set of symbols, all different, forming a representation of the said first arrangement and a second set of symbols, all different, forming a representation of a second arrangement of words, and

* an operator (8) devised so as to receive as input a primary elementary data item, as well as the said first and second sets of symbols, and so as to perform on this primary elementary data item, word transformations defined solely by the said first and second sets of symbols in such a way as to output a corresponding secondary data item equivalent to the said primary elementary data item.

2. Device according to Claim 1, in which a first item of equipment (1) delivers the said primary elementary data items coded according to the said first arrangement, and a means of conversion (5') delivers secondary data items coded according to a third arrangement, after conversion of primary elementary data items coded according to a fourth arrangement by a second item of equipment (1'), the said items of equipment (1, 1') desiring to exchange primary elementary data items,

characterized in that the said operator (8) comprises means of interrogation (9) devised so as:

* to supply the said second item of equipment (1′) with a message containing the said second set of symbols and requiring the sending back to the said operator (8) of a primary elementary data item, transform of the said second set of symbols by coding according to the said fourth arrangement,

* deducing from this primary elementary data item as well as from the first and second sets of symbols a third set of symbols forming a representation of the said fourth arrangement,

* replacing the said second set of symbols by the said third set of symbols, in the said operator (8) and in the said means of conversion (5′), so that in the event of the transmission of a primary elementary data item coded according to the first, respectively fourth, arrangement and intended for the said second, respectively first, item of equipment, the said operator (8) delivers to the latter, directly, a primary elementary data item coded according to the fourth, respectively first arrangement.

3. Device according to Claim 2, characterized in that the said first, second and third sets of symbols are ordered strings of numbers.

4. Device according to Claim 3, characterized in that the said second set of symbols is the string [1, 2, 3,..., n-1, n], n being the number of

components of a base over which the secondary data item is decomposed into words of k bits, k being greater than or equal to 1, and in particular equal to 8.

5. Device according to one of Claims 2 to 4, characterized in that the said second and third arrangements are identical.

6. Device according to one of Claims 2 to 4, characterized in that the said second and third arrangements are different and are associated with sets of symbols comprising numbers of words which are different and/or words of number of bits which are different, and in that the said means of interrogation are devised so as to address to the said means of conversion (5') at least one second set of symbols so that it is substituted for the said third arrangement.

7. Device according to Claim 6, characterized in that the said storage means (7) store several second sets of symbols comprising numbers n_i of words which are different and/or of words of number k_i of bits which are different, and in that the said means of interrogation (9) are devised so as to supply the said second item of equipment (1'), via a message, with a chosen number of first sets of symbols which are different.

8. Device according to one of Claims 2 to 7, in which the said first item of equipment (1) is stored in a first machine (M1), in particular a computer, characterized in that the said storage means

(7) and a part at least of the said operator (8, 9-1) are installed in the said first machine.

9. Device according to Claim 8, characterized in that the said storage means (7) and a part (8, 9-1) at least of the said operator are installed in the form of a program in the said first machine (M1).

10. Device according to one of Claims 8 and 9, in which the said second item of equipment (1') and the said means of conversion (5) are installed in a second machine (M2), in particular a computer, characterized in that a first part (9-1) of the said interrogation means is installed in the said first machine, while a complementary second part (9-2) is installed in the said second machine (M2).

11. Device according to Claim 10, characterized in that the said means of interrogation (9-2) are installed in the form of a program.

12. Device according to Claim 11, characterized in that the said operator (8) is devised so as to install the said second part (9-2) of the means of interrogation in the said second machine (M2) when the said first item of equipment attempts, for the first time, to exchange primary elementary data items with the said second item of equipment (1').

13. Process for converting primary elementary data items individually coded according to a first arrangement of words, first arrangement of words,

characterized in that it comprises the following steps:
a) providing a first set of symbols, all different, forming a representation of the said first arrangement and a second set of symbols, all different, forming a representation of a second arrangement of words and,
b) receiving a primary elementary data item, as well as the said first and second sets of symbols, and
c) performing on this primary elementary data item, word transformations defined solely by the said first and second sets of symbols in such a way as to output a corresponding secondary data item equivalent to the said primary elementary data item.

14. Process according to Claim 13, in which a primary elementary data item coded according to the said first arrangement is received from a first item of equipment and a secondary data item coded according to a third arrangement is received from a means of conversion, the latter data item arising from the conversion of a primary elementary data item coded according to a fourth arrangement by a second item of equipment,
characterized in that in step b):
* the said second item of equipment is supplied with a message containing the said second set of symbols and requiring the sending back of a primary elementary data item, transform of the said first set of symbols by coding according to the said fourth arrangement, then

* a third set of symbols forming a representation of the said fourth arrangement is deduced from this primary elementary data item as well as from the first and second sets of symbols, and

* the said second set of symbols is replaced everywhere by the said third set of symbols, so that in the event of the transmission of a primary elementary data item coded according to the first, respectively fourth, arrangement and intended for the said second, respectively first, item of equipment, a primary elementary data item coded according to the fourth, respectively first arrangement is delivered directly to the latter item of equipment.

1/2

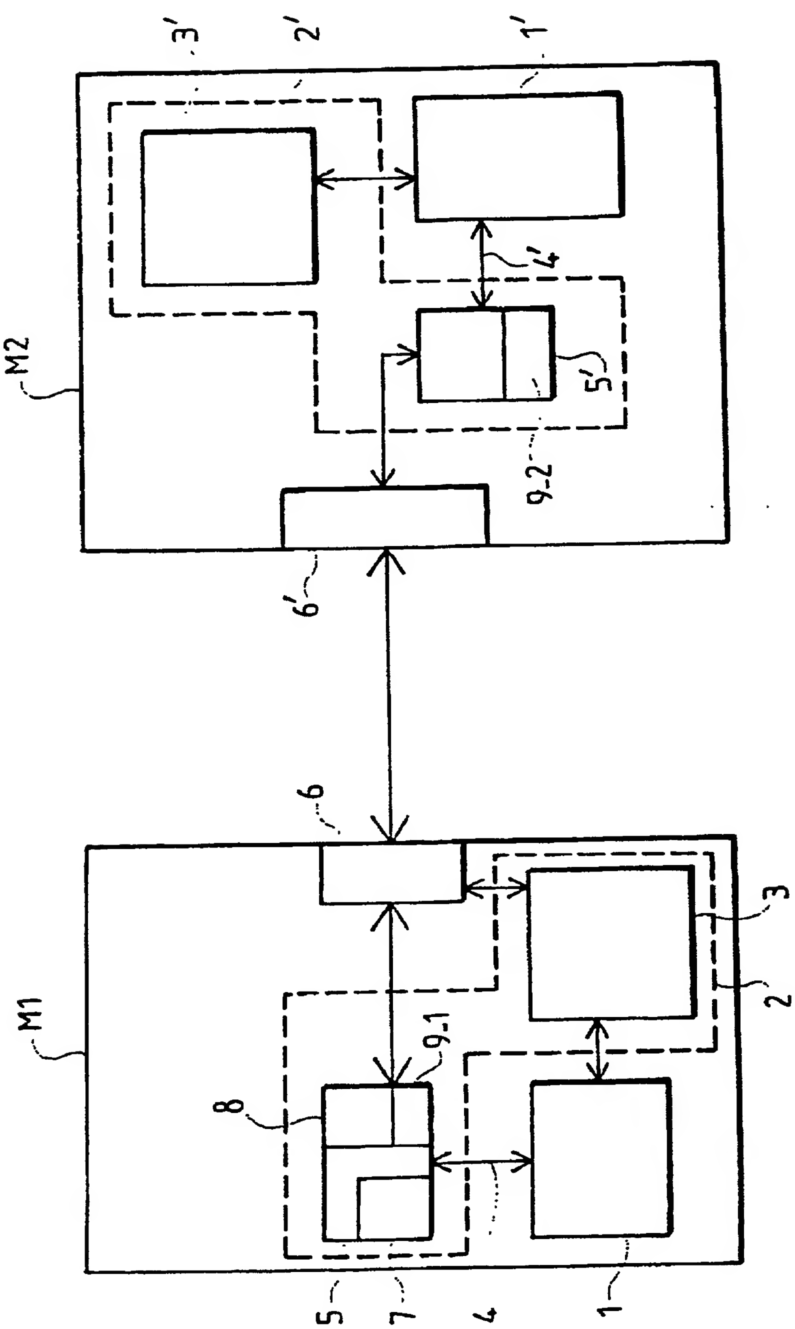

FIG.1

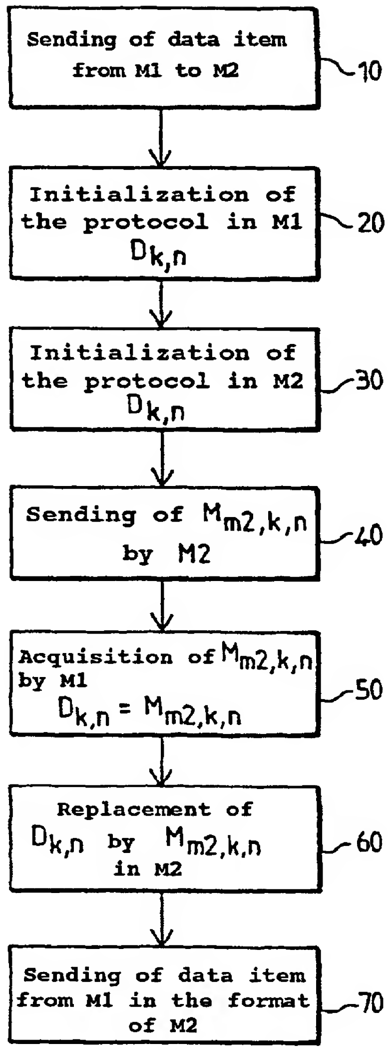
Sending of data item from M1 to M2
10
Initialization of the protocol in M1 $D_{k,n}$
20
Initialization of the protocol in M2 $D_{k,n}$
30
Sending of $M_{m2,k,n}$ by M2
40
Acquisition of $M_{m2,k,n}$ by M1 $D_{k,n} = M_{m2,k,n}$
50
Replacement of $D_{k,n}$ by $M_{m2,k,n}$ in M2
60
Sending of data item from M1 in the format of M2
70

FIG. 2

Docket No. ______________

DECLARATION AND POWER OF ATTORNEY

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name.

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled:

the specification of which is attached hereto unless the following box is checked:

☒ was filed on June 23, 2000 as United States Application Number or PCT International Application Number PCT/FR00/01767 and was amended on ______________ (if applicable).

I hereby state that I have reviewed and understand the contents of the above-identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is known by me to be material to patentability as defined in Title 37, Code of Federal Regulations § 1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, § 119(a)-(d) or § 365(b) of any foreign application(s) for patent or inventor's certificate, or § 365(a) of any PCT International application which designated at least one country other than the United States, listed below and have also identified below any foreign application for patent or inventor's certificate, or PCT International application having a filing date before that of the application on which priority is claimed:

PRIOR FOREIGN APPLICATION(S)

NUMBER	COUNTRY	DAY/MONTH/YEAR FILED	PRIORITY CLAIMED
99/08172	FRANCE	25/06/1999	YES

I hereby claim the benefit under Title 35, United States Code § 119(e) of any United States provisional application(s) listed below.

APPLICATION NO.	FILING DATE

I hereby claim the benefit under Title 35, United States Code, § 120 of any United States application(s), or § 365(c) of any PCT International application designating the United States, listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States or PCT International application in the manner provided by the first paragraph of Title 35, United States Code, § 112, I acknowledge the duty to disclose information which is known by me to be material to patentability as defined in Title 37, Code of Federal Regulations § 1.56 which became available between the filing date of the prior application and the national or PCT International filing date of this application:

APPLICATION SERIAL NO.	FILING DATE	STATUS: PATENTED, PENDING, ABANDONED

I hereby appoint as my attorneys, with full powers of substitution and revocation, to prosecute this application and transact all business in the Patent and Trademark Office connected therewith: Stephen A. Bent, Reg. No. 29,768; David A. Blumenthal, Reg. No. 26,257; William T. Ellis, Reg. No. 26,874; John J. Feldhaus, Reg. No. 28,822; Patricia D. Granados, Reg. No. 33,683; John P. Isacson, Reg. No. 33,715; Donald D. Jeffery, Reg. No. 19,980; Eugene M. Lee, Reg. No. 32,039; Richard Linn, Reg. No. ~~25,144~~; Peter G. Mack, Reg. No. ~~26,001~~; Brian J. McNamara, Reg. No. 32,789; Sybil Meloy, Reg. No. 22,749; George E. Quillin, Reg. No. 32,792; Colin G. Sandercock, Reg. No. 31,298; Bernhard D. Saxe, Reg. No. 28,665; Charles F. Schill, Reg. No. 27,590; Richard L. Schwaab, Reg. No. ~~25,479~~; Arthur Schwartz, Reg. No. 22,115; Harold C. Wegner, Reg. No. 25,258.

19

Docket No.________________

Address all correspondence to FOLEY & LARDNER, Washington Harbour, 3000 K Street, N.W., Suite 500, P.O. Box 25696, Washington, D.C. 20007-8696. Address telephone communications to ________________ ~~at (202) 672-5300~~.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

1-00

Full Name of First or Sole Inventor	Signature of First or Sole Inventor	Date
ROUILLIER Fabrice	Fabrice Rouillier	Jan 14 2002
Residence Address	**Country of Citizenship**	
8, rue des Bonnetiers, F -60200 COMPIEGNE FRANCE FRX	FRANCE	
Post Office Address		
same address		

2-00

Full Name of Second Inventor	Signature of Second Inventor	Date
FAUGERE Jean-Charles	Jean-Charles Faugère	Jan 14 2002
Residence Address	**Country of Citizenship**	
12, Avenue Parmentier F-75011 PARIS (FRANCE) FRX	FRANCE	
Post Office Address		
same address		

Full Name of Third Inventor	Signature of Third Inventor	Date
Residence Address	**Country of Citizenship**	
Post Office Address		

Full Name of Fourth Inventor	Signature of Fourth Inventor	Date
Residence Address	**Country of Citizenship**	
Post Office Address		

Full Name of Fifth Inventor	Signature of Fifth Inventor	Date
Residence Address	**Country of Citizenship**	
Post Office Address		